Python Beginners Cohort 25 Project

Mock E-Commerce Application

# Project Overview

The purpose of this project is to design and implement a mock e-commerce application utilizing the Python programming language. Participants are required to develop a console-based application that replicates the essential functionalities of a basic e-commerce platform. This assignment aims to provide an opportunity for participants to demonstrate their proficiency in Python programming concepts as well as their ability to structure and develop a practical application.

# Project Requirements

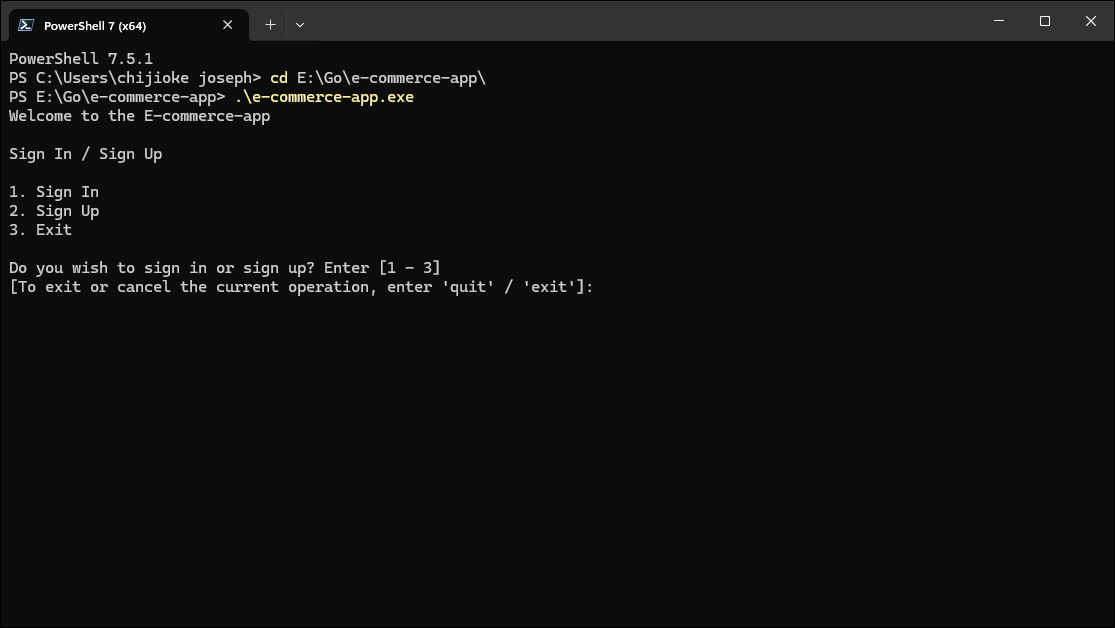
* The application must be implemented exclusively in Python.
* The application should operate through the console interface, with no requirement for a graphical user interface (GUI).
* The application must incorporate features representative of core e-commerce functionalities, including but not limited to product listing, user interaction, and order processing.

# Guidelines

* Participants are expected to organize their code into coherent functions to enhance readability and maintainability.
* The program should effectively manage user inputs and provide clear and meaningful responses or feedback.
* Participants should ensure the application is robust by implementing mechanisms to handle potential errors gracefully, such as invalid inputs.
* You have also been provided with an executable that you can use to check if you are heading in the right direction. Please if you wish to run the executable, follow these instructions:
  + Create a new folder
  + Place the executable inside that folder
  + Create a ‘data’ folder inside that folder
  + Place the ‘accounts.log’ and ‘warehouse\*.txt’ files that you are provided in that data folder.
  + Navigate to that new folder in your computer terminal and run the following command

.\e-commerce-app.exe

* See the search.py file attached for how to search through inventory items in your program.



# Features

## Storage

### Warehouse Storage

The application is equipped with multiple text files named according to the pattern warehouse\*.txt. These files are to be stored within the data directory. The program is required to process these files to construct a data structure representing all items available for sale in the application.

Each item listed within the warehouse\*.txt files comply with the following format:

<NAME>:<PRICE>

* The <NAME> denotes the item’s designation.
* The <PRICE> specifies the selling price in Nigerian Naira (NGN).
* Items within the file are delimited by a semicolon (;).
* The <NAME> and <PRICE> within each item entry are separated by a colon (:).

The program must implement robust parsing methods to accurately interpret and store the contents of these files into a structured data format, ensuring all items are seamlessly integrated into the application’s inventory system.

### Profile Storage

To facilitate the management of user profiles, the program is required to implement a storage system aligned with the following specifications:

* Within the data directory, a file titled accounts.txt should be established to store user account information.
* At runtime, the program must verify the existence of the data directory:
* If the data directory does not exist, the program should create it automatically.
* If the data directory exists but the accounts.txt file is absent, the program should generate the file.
* If both the data directory and accounts.txt file are present, the program should proceed without modification.

The accounts.txt file will act as a repository to systematically store user profile data, ensuring ease of retrieval and manipulation in future operations.

## Login

The program begins by displaying a Login section to manage user access, offering two options: Sign-In and Sign-Up. Upon execution, the user is prompted to either sign in or sign up to proceed.

### Sign-In

For returning users, the Sign-In option will request the following information:

* Username or Email: The user must provide either their username or email address.

The program will verify that the provided username or email exists in the accounts.txt file. If it does, the user will be prompted to enter their password:

* Password: The user must provide their password, which must match the one stored in the accounts.txt file for the corresponding username or email.

If the credentials are valid, the user gains access to the application. If the credentials are invalid, an error message is displayed, and the user is prompted to re-enter their details or choose an alternative option.

The program should read the user’s credentials and then load the user’s details from the account.txt file. The account.txt file should have the user’s details stored in this format:

<USERNAME>, <EMAIL>, <PASSWORD>, <BALANCE>.

Each detail should be stored on a line with each detail separated by a comma (,).

### Sign-Up

For new users, the Sign-Up option guides them through creating a new account. The program will require:

* Username: A unique identifier that does not duplicate existing entries in accounts.txt.
* Email: A unique email address that has not been used by any existing account.

Users will then choose one of two options for creating a password:

* Manual Entry: The user can enter a password manually, but it must meet the following criteria:
* At least 16 characters long.
* Contains at least one lowercase letter, one uppercase letter, one number, and one special symbol.

Automatic Generation: The program will generate a password that complies with the same rules as above.

Once the account is successfully created, the program will store the user's details in the accounts.txt file in the following format:

<USERNAME>, <EMAIL>, <PASSWORD>, <BALANCE>

The balance is set to NGN 0.00 by default at the time of account creation.

By consolidating Sign-In and Sign-Up under the Login section, the program ensures a streamlined and secure experience while maintaining strict compliance with user data standards.

### Exit

This subsection allows users to terminate the program should they choose not to proceed with Sign-In or Sign-Up. By selecting this option, the program will safely exit without making any changes or storing any information.

## Run

The Run Section is central to the program’s functionality, enabling users to perform key actions seamlessly. It is designed with four main functionalities to cater to diverse user needs:

* Funding a User’s Wallet: This feature allows users to add money to their wallet securely. Users can specify the amount they wish to deposit, after which the system updates their balance accordingly. Transactions are processed efficiently while maintaining data integrity.
* Making Purchases: Users can utilize their wallet balance to make purchases directly within the program. This feature ensures that transactions are smooth and keeps track of purchase history for user reference. It also alerts users if their balance is insufficient, prompting them to fund their wallet.
* Managing a User’s Account: Account management empowers users to update personal details such as their username or email, as well as view or change their password. The feature ensures user data is kept up-to-date and enables convenient management without compromising security.
* Exiting the Program: For users wishing to terminate their session, this option ensures a safe and clean exit. Selecting this functionality closes the program without retaining unsaved changes or sensitive information, preserving a secure and efficient user experience.

By leveraging these functionalities, the Run Section guarantees a comprehensive and user-friendly experience, making the program both practical and intuitive for all.

### Fund

The Fund feature enables users to securely add money to their wallet through an interactive and straightforward process. Upon selecting this option within the Run Section, users are guided through the following steps:

* Selection of Amount: The program presents users with a set of fixed funding options (e.g., NGN 10,000, NGN 20,000, NGN 50,000, NGN 100,000) numbered for convenience. Users choose the desired amount by entering the corresponding option number.
* Account Increment: Based on the user’s selection, the program increments the user’s wallet balance by the specified amount.
* Saving to Record: The updated balance is then securely saved to the user's record in the “accounts.txt” file located in the “data” folder. This ensures the integrity and accuracy of user account data.
* Next Steps: After successfully funding the wallet, the program prompts the user to decide whether they wish to continue funding their account or return to the previous menu. If the user opts to continue funding, the process repeats; if they choose to return, the program navigates back to the Run Section menu seamlessly.

### Purchase

The Purchase section is a dedicated part of the program that allows users to interact with the store, manage their shopping cart, and complete transactions. It provides an intuitive and streamlined experience with clear menu options for efficient navigation.

When accessing the Purchase section, users are presented with the following menu options:

* Search: This option lets users search for items in the store by entering a query string. The system returns matching results for easy browsing and selection.
* Manage Cart: Provides users with tools to view, add, remove, and clear items from their shopping cart, ensuring a flexible and organized shopping experience.
* Checkout: Allows users to review their cart, view the total checkout price, and confirm payment. If the wallet balance is insufficient, the payment is rejected, and users are returned to the Purchase menu.
* Exit Purchase Menu: Exits the Purchase section and redirects users back to the main menu in the Run Section.

The Purchase section ensures users can confidently explore and manage their shopping needs while maintaining a seamless connection with the rest of the program.

#### Search

The "Search" feature prompts the user to input the name or brand of the item they wish to purchase. This input is processed to ensure case-insensitive matching and allows partial searches for improved flexibility. The system then uses the user's query to find relevant items in the program's inventory.

##### Algorithm

The process consists of the following steps:

* Prompt User Input: The user is prompted to enter a search query (e.g., the name or brand of an item).
* Break Query into Words: The program splits the input string into individual words for detailed matching.
* Create Regex Patterns: A case-insensitive regular expression (regex) is created for each word from the query.
* Loop Through Inventory: The program iterates through all item names in the inventory, applying each regex pattern to the item names.
* Store Matching Items: An item is stored in the search output only if its name matches all regex patterns created from each word in the query.
* Display Results: The list of matched items is presented to the user for further action.

##### Example Code

Below is an example implementation of this algorithm in Python:

```python

import re

# Example inventory

inventory: list[str] = [

"Apple iPhone 14",

"Samsung Galaxy S23",

"Google Pixel 7",

"Sony Xperia 5",

"Apple Watch Series 8",

"Samsung Smart TV",

]

```

```python

def search\_inventory(query: str, inventory: list[str]) -> list[str]:

# Step 1: Split the user query into individual words

search\_terms: list[str] = query.split()

# Step 2: Create case-insensitive regex patterns for each word

regex\_patterns: list[re.Pattern[str]] = [re.compile(re.escape(term), re.IGNORECASE) for term in search\_terms]

# Step 3: Filter inventory items based on the regex patterns

search\_output: list[str] = []

for item in inventory:

if all(pattern.search(item) for pattern in regex\_patterns):

search\_output.append(item)

# Step 4: Return matched items

return search\_output

```python

```python

# Example Usage

user\_query = "Apple Watch"

matched\_items = search\_inventory(user\_query, inventory)

print("Matched Items:")

print(matched\_items)

```

##### Explanation of Code

* Inventory: A list of item names represents the inventory.
* Query Splitting: The user's input is split into individual words using the `split()` method.
* Regex Creation: For each search term, a regex pattern is created using `re.compile()` with the `re.IGNORECASE` flag to ensure case-insensitivity.
* Matching Logic: The program loops through the inventory and checks if all regex patterns match the item's name using `all()`.
* Output: Matched items are appended to `search\_output`, which is returned and displayed to the user.

##### Example Output

If the user queries "Apple Watch," the program outputs:

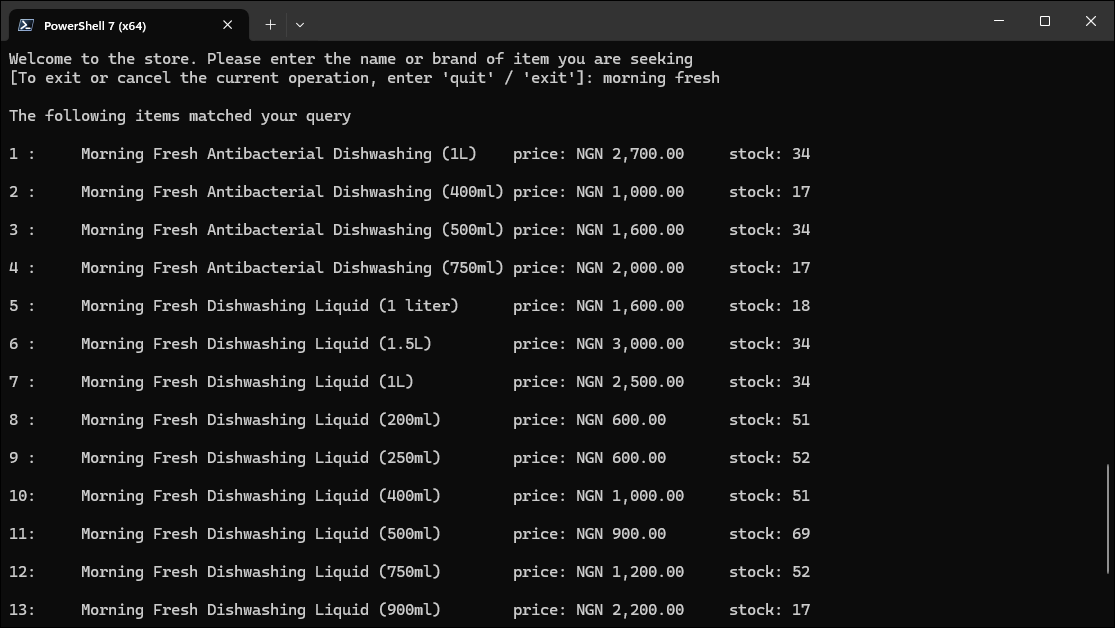
```

Matched Items:

['Apple Watch Series 8']

```

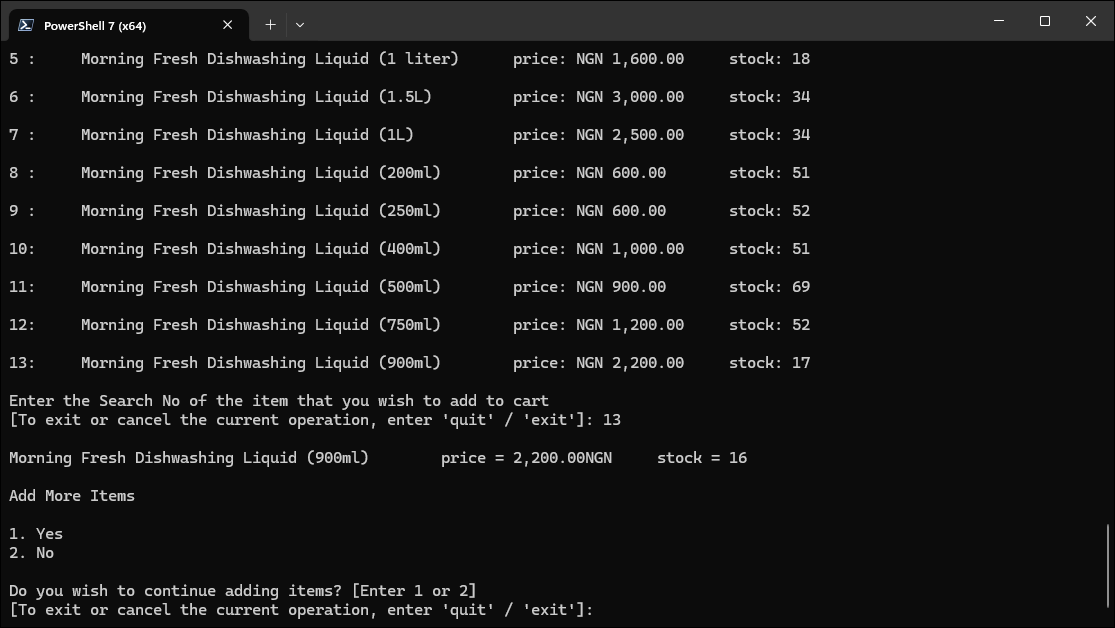
An example of how the search output is displayed is shown in the image below.



To improve user interaction and provide a detailed and user-friendly experience, an additional menu is introduced after displaying the matched items. This menu includes key options, each outlined with specific functionality:

* Search Again: This option allows users to refine or modify their search queries. Upon selecting this, the program reinitiates the entire search process, enabling users to explore the inventory further or correct any earlier input errors. It ensures that users have an iterative and flexible search experience, catering to diverse needs and preferences. Additionally, if no items match the user’s query, the program should display a message informing the user that no items have matched their input.
* Add Items to Cart: When this option is selected, the program re-displays the search results, listing each item with a corresponding number for easy identification. Users can add an item to their cart by entering the assigned number. Once an item is added to the cart:
  + The stock or quantity of the item in the inventory is decreased by one to reflect the addition.
  + The program simultaneously increases the quantity of the same item in the user's cart, ensuring accurate record-keeping across both systems.

This functionality ensures real-time inventory updates and prevents any discrepancies between the inventory and the user’s cart.



* Exit Search Menu: Choosing this option immediately returns the user to the Purchase Menu, allowing them to seamlessly transition to other purchasing actions or functionalities. This ensures that users can continue their shopping journey without unnecessary delays or complications.

This systematic approach enhances the user's ability to find items efficiently and intuitively, ensuring a seamless shopping experience.

#### Manage Cart

Manage Cart: This option provides users with a comprehensive set of functionalities for managing the items in their cart. Upon selecting this menu, the program displays several options tailored to ensure flexibility and control over the shopping experience. The detailed functionalities included are:

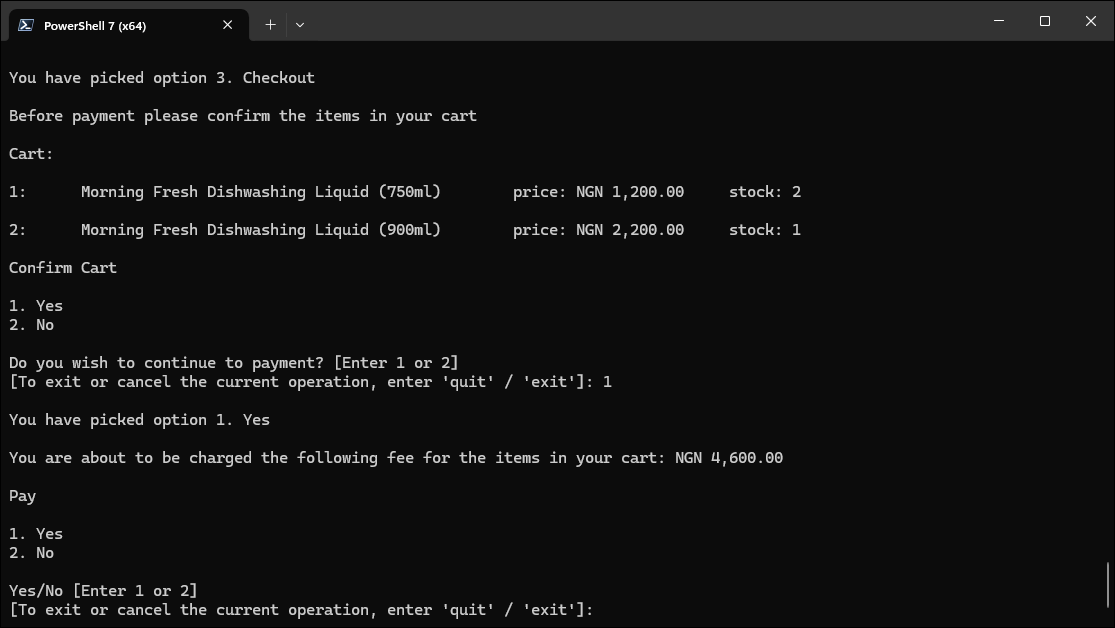
* View Items in Cart: This option allows users to examine the current contents of their cart. The program presents a detailed list displaying the name, quantity, and individual price of each item in the cart. Users can use this feature to verify their selections and make informed decisions before proceeding to a transaction or modifying their cart.
* Add Items to Cart: Users can increase the quantity of specific items in their cart by selecting this option. The program re-displays the inventory alongside the item numbers, enabling users to conveniently select and add their desired products. As items are added, the inventory quantity is adjusted in real time to reflect the change, ensuring accuracy and preventing overselling.
* Remove Items from Cart: This functionality empowers users to decrease the quantity of specific items in their cart or remove them entirely. The program lists all items currently in the cart with corresponding identification numbers. Users may input the number of the item they wish to modify or remove, and the program updates both the cart and the inventory accordingly.
* Clear Cart: Selecting this option allows users to empty their cart entirely, removing all items at once. This is particularly useful if users wish to start over or cancel their selections for any reason. The inventory is updated to restore the quantities of the cleared items, ensuring consistency in stock management.
* Exit Manage Cart Menu: This option allows users to exit the Manage Cart menu and return to the previous menu seamlessly. This ensures that users can continue their interaction with the program without any unnecessary interruptions or complications.

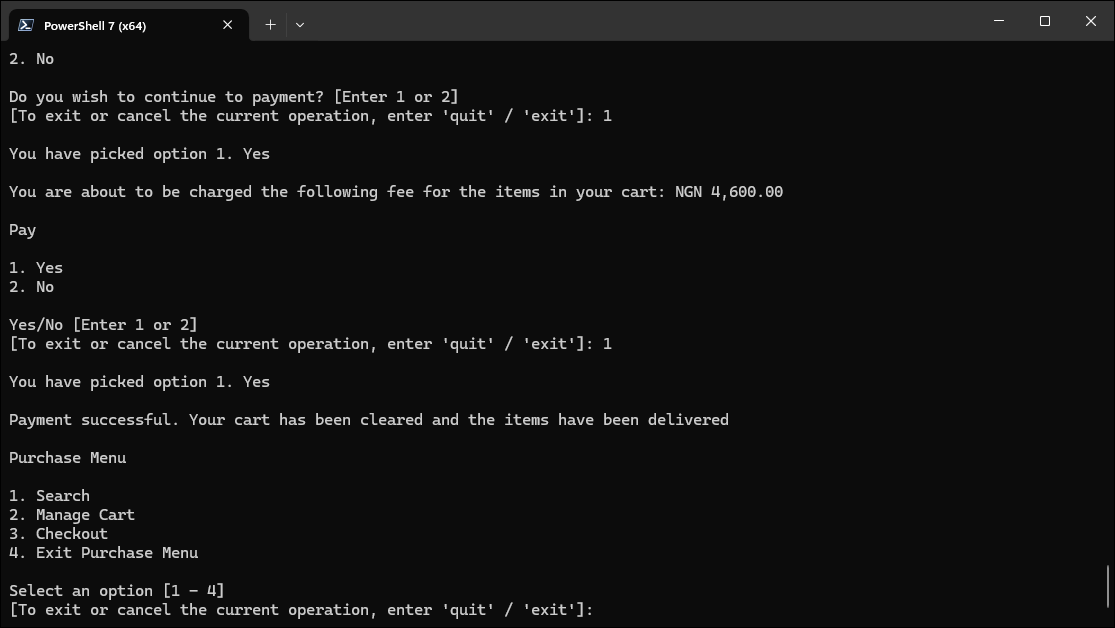
#### Checkout

The Checkout option allows users to review their cart, confirm payment details, and proceed with a secure transaction. The program calculates the total fee, verifies funds, and debits the user's account if sufficient funds are available, ensuring error handling for insufficient balances. A success message is displayed after the transaction, and the user is returned to the previous menu.

* Cart Display: The program shows the contents of the user's cart, including items, quantities, and individual prices.
* Confirmation Prompt: Users are asked if they wish to proceed to payment or return to the previous menu.
* Total Fee Calculation: If the user opts to proceed, the program calculates and displays the total fee for all items in the cart.
* Payment Confirmation: Users confirm their intent to pay the displayed total fee.
* Fund Verification: The program checks the user's account balance to ensure there are sufficient funds.
* Error Handling: If funds are insufficient, an error message is displayed, and the user is returned to the previous menu without completing the transaction.
* Transaction Processing: If sufficient funds are available, the program debits the fee from the user's account and updates the account records securely.
* Success Message: The program prints a message confirming the transaction was successful.
* Return to Previous Menu: After the transaction, the user is seamlessly returned to the previous menu.

An example of all these steps is provided in the image below:





#### Exit Purchase Menu

This option is designed to streamline user navigation within the program. After completing the purchase process, users are effortlessly redirected to the main menu under the Run section, ensuring a smooth transition without disruptions or unnecessary steps.

### Manage Account

The Manage Account section is designed with security and data integrity as top priorities. It has the following features:

#### Verification

For most operations within this section, users are required to verify their identity by entering their password. This added layer of security ensures that sensitive actions, such as changing account details or resetting balances, are initiated exclusively by authorized individuals. For example, if a user wishes to update their email address, they must first authenticate themselves by providing their password, preventing unauthorized access or unintended changes.

#### Data Persistence

Another key feature of the Manage Account section is its robust mechanism for data persistence. Any updates made to user information, whether it is a change in username, email, password, or balance, are instantly and securely saved to the “accounts.txt” file located in the “data” folder. This ensures that the program maintains a consistent and accurate record of all account modifications. For instance, when a user resets their balance, the program immediately reflects this change in the “accounts.txt” file, safeguarding the integrity of the account data and minimizing discrepancies.

#### Confirmation for Sensitive Operations

In addition to password verification, certain operations require an extra confirmatory prompt to ensure the user’s intent. This feature is applied particularly for sensitive actions like resetting balances or deleting accounts. For example, when a user opts to delete their account, they are presented with a confirmation message asking them to confirm their decision. Only after the user explicitly agrees does the program proceed to delete the account and its corresponding record from the “accounts.txt” file. This double-layered verification prevents accidental deletion and reinforces security.

The integration of these features — identity verification, immediate data updates, and confirmatory prompts — demonstrates the program's commitment to providing users with a secure and seamless account management experience.

The Manage Account section under the Run section allows users to manage their accounts through a menu with the following options:

* Change Username: Allows the user to update their current username. The user must verify their identity by entering their password before the operation proceeds. The change is updated in the “accounts.txt” file.
* Change Email: Permits the user to modify the email associated with their account. Identity verification via password is required before making this change. The updated email is saved to the “accounts.txt” file.
* Change Password: Enables the user to set a new password for their account. The existing password must be entered for verification. Once completed, the new password is recorded in the “accounts.txt” file.
* View Account Balance/Details: Displays the user's account balance and other account-specific details. The operation requires the user to verify their identity using their password.
* Reset Balance: Resets the user's wallet balance to zero. This operation requires both password verification and an extra confirmatory prompt where the user must assent to proceed. Upon completion, the change is reflected in the “accounts.txt” file.
* Delete Account: Permanently removes the user’s account and its associated record from the “accounts.txt” file. This action requires password verification followed by an additional confirmatory prompt. Once confirmed, the account is deleted, and the user is redirected to the initial prompt page in the Login section.
* Logout: Logs the user out of their account and redirects them to the initial prompt page in the Login section.
* Exit Manage Account: Exits the Manage Account section and redirects the user back to the main menu in the Run section.

### Exit

The Exit feature is designed to provide users with a seamless and secure conclusion to their session. It performs multiple functions to ensure a smooth exit from the program. Before the program closes, the user is logged out of their account securely, safeguarding their data. Additionally, a farewell message is displayed, which is crafted to leave a positive impression, thanking the user for their visit and encouraging them to return. This ensures that the user experience is complete and reassuring, even at the point of departure.

* Logs the user out of their account securely before exiting the program.
* Displays a customizable farewell message tailored for commercial sites, thanking the user and inviting them back.
* Confirms to the user that their logout process is complete, enhancing their sense of security.

# Presentation

## Date

The presentation date is scheduled for Wednesday 4th June 2025 at 9:00am prompt. This is still subject to change and any such change will be communicated accordingly on the WhatsApp Group. The timing of the presentation is very important. Any group that delays the presentation will automatically fail.

## Rules

* Every group is expected to send three (3) representatives which must be both include both Male and Female reps.
* The Group Leader and Assistant Leaders are not required to present, and the ruling goes that both of the leaders cannot represent the group. The minimum allowed is one (1) group leader.
* No Slides are required for your presentation, just your codes and your representatives who will explain the program that you have built.

## Submission

* Group submission is done on the morning of the presentation. You are expected to submit your Python file named as follows:

“**Group 1**” with the zip file extension for members of Group 1. The name must match the casing as well.

* Your zip file should contain the python files for your project. The python files should be properly named. Your group file should also contain a “**Markdown**” file which explains briefly what your project is all about. You can also go ahead to list the members of the group as well for posterity’s sake.

That said. I wish you guys good luck 🤞 in your project. Please if you have any questions consult ChatGPT or Stack Overflow. You can also refer to the executable which I will be sending to you as well. It can only be run on Windows.